从0到1到无穷

vivo大规模机器学习实践

本次演讲的要点

- 如何快速的在企业中快速实现大规模机器学习算法端到端落地
- 由简单离线系统向复杂实时系统的演进
- 在算法迭代中的经验和教训
- 对未来的展望

vivo互联网算法相关业务介绍

初创团队普遍存在的问题

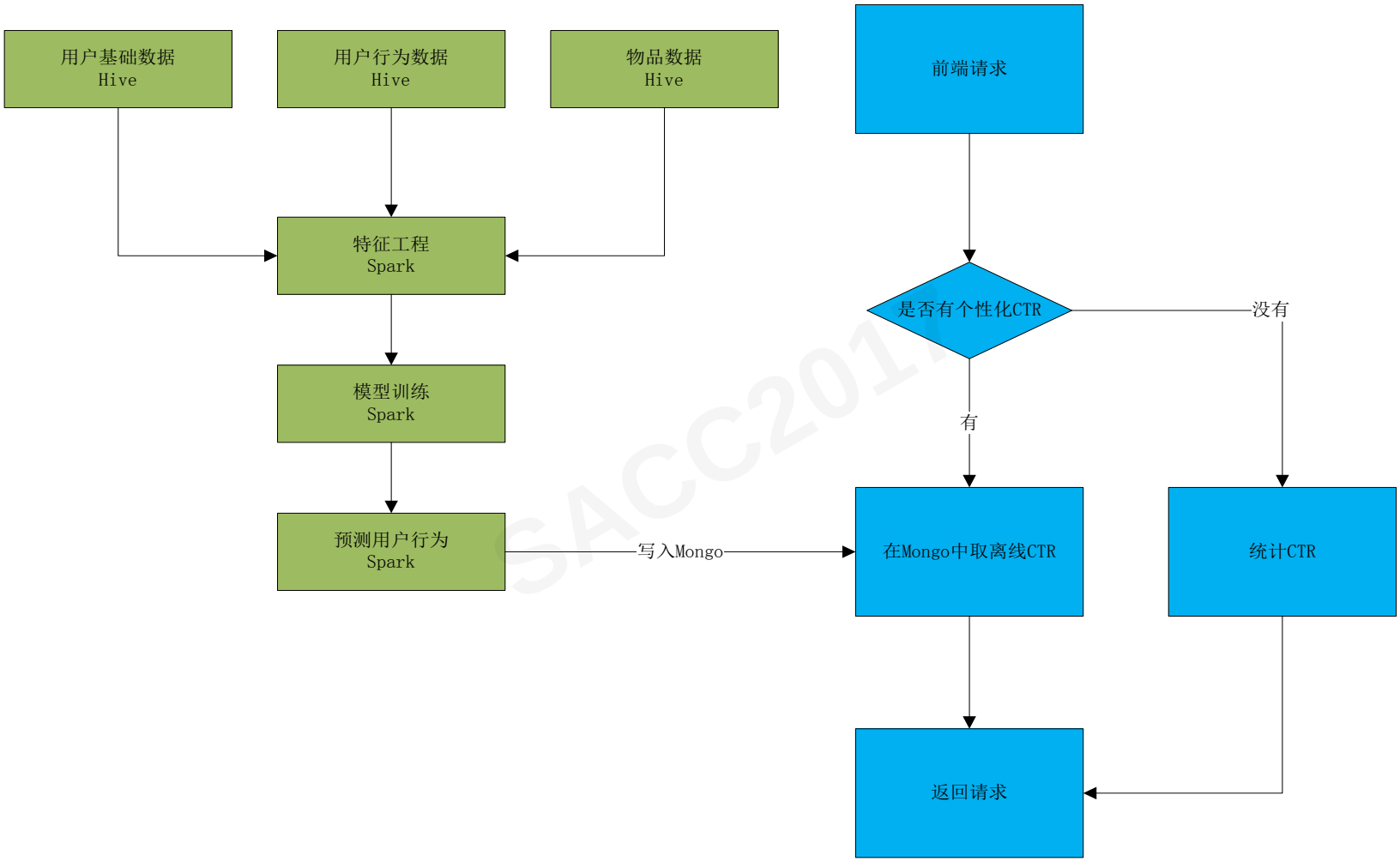
1.两三条枪

2.数据中坑很多

3. 技术积累弱

4.领导希望尽快看到效果

第一代解决方案架构-2016



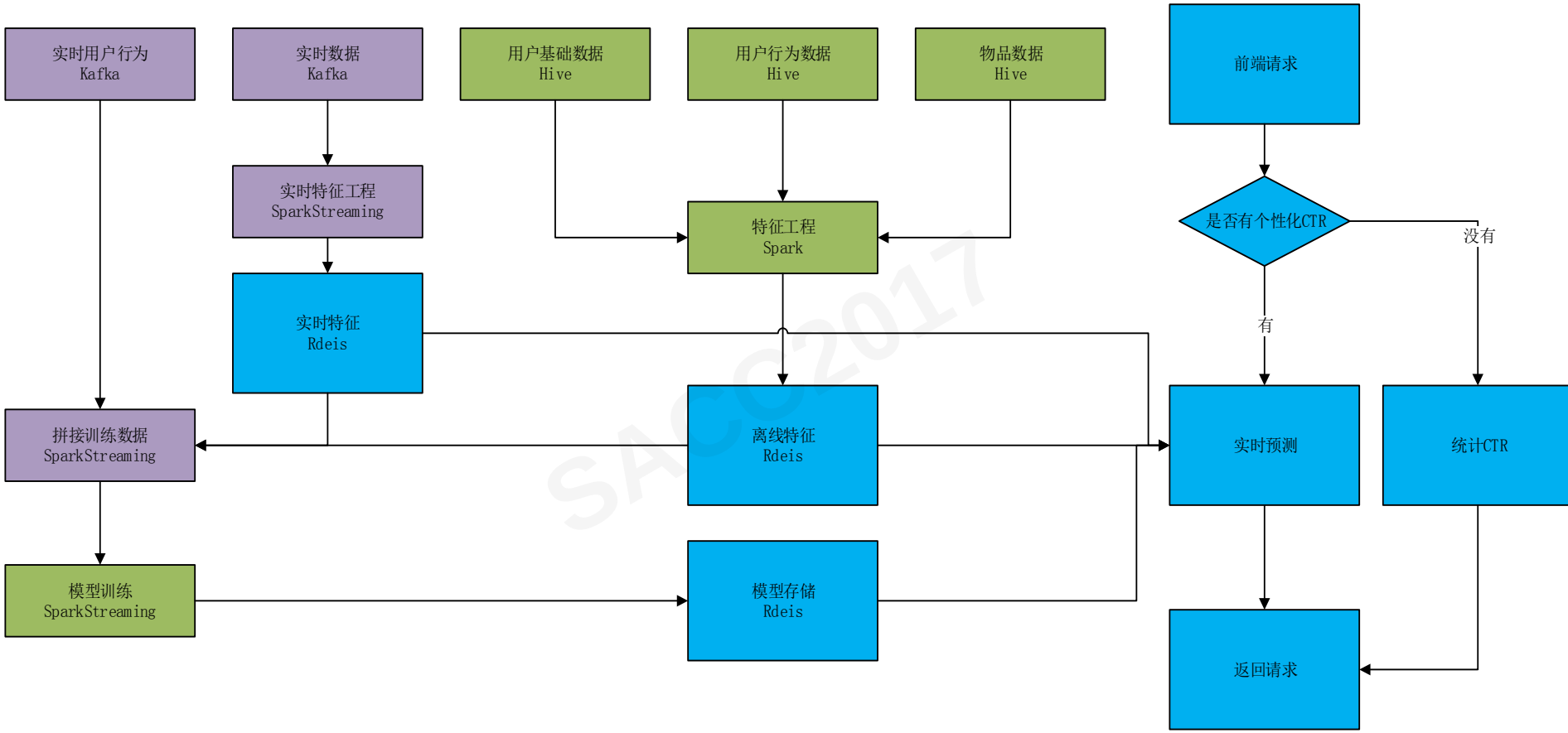
第一代解决方案的优劣势

- **优势：**
  - 对算法团队技能要求单一
  - 很好的利用现有大数据架构
  - 对工程团队要求低
  - 出错的几率小

- **劣势：**
  - 1. 离线预测，很多实时特征用不了
  - 2. 离线训练，模型更新较慢
  - 3. 用Spark进行训练，可选模型少，效率低，训练数据的规模有瓶颈

- **接入业务：**
  - APP展示广告，游戏推荐

第二代解决方案架构-2017上半年



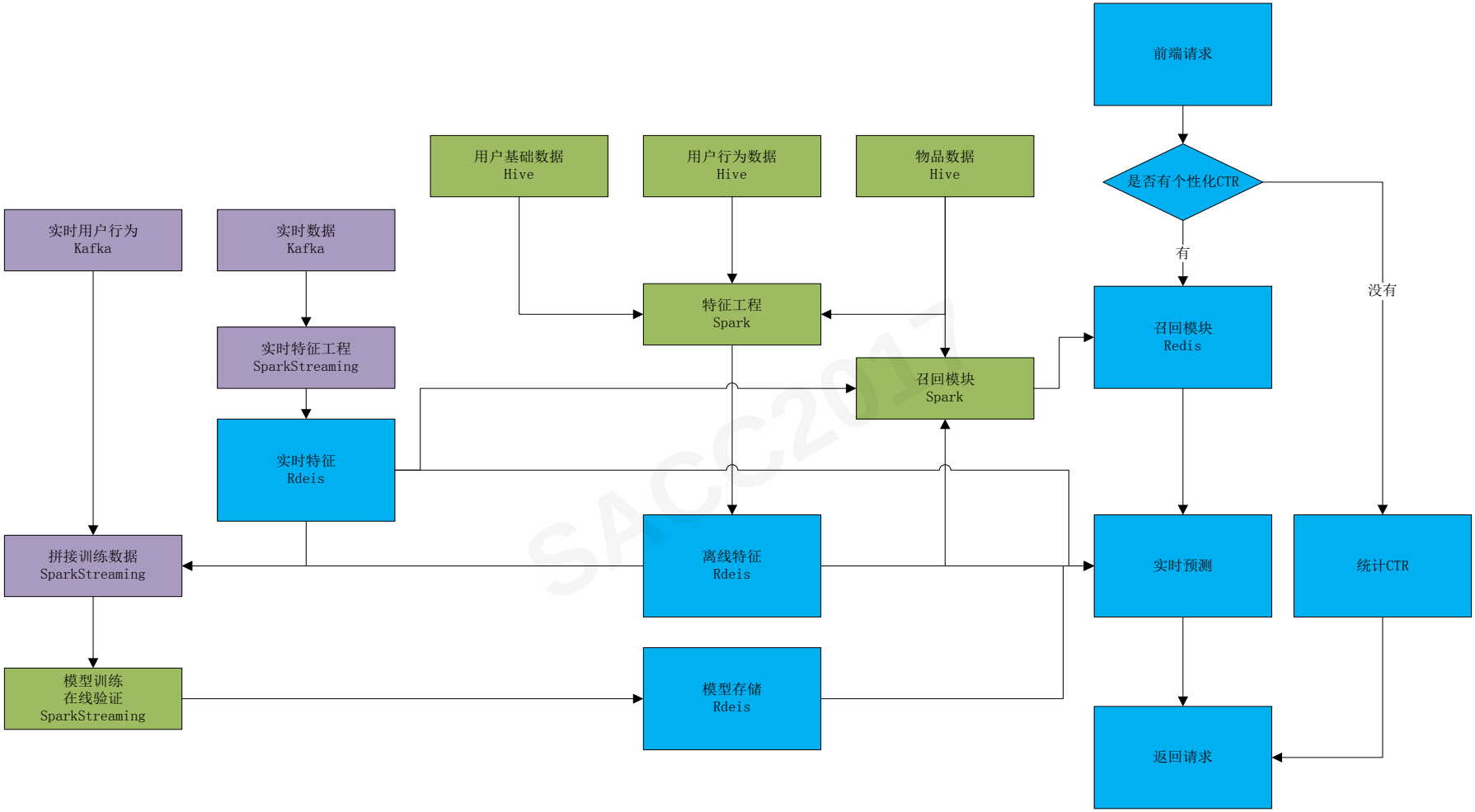
第二代解决方案解决的进步和挑战

- **进步：**
  - 实时预测，能够使用上下文，时间等场景信息 。
  - 在线训练，能够学习新广告，适应概念漂移。
- **挑战：**
  - 大量使用实时数据，工程端承担线上预测部分开发，出错的可能性增大
  - 算法迭代涉及大数据和工程改动，成本高周期长
- **接入业务：**
  - 信息流广告，APP推荐

第二代解决方案的经验

- 实时特征工程一定要存原始数据
- 特征交叉，甚至一部分特征工程用jar包的方式热加载
- 实时预测模块要打日志，包括读取的数据和预测结果
- 客户端上传日志的时候要透传预测的CTR和requestID
- 模型要先做线下验证，不光是整体的，还要单个item的
- 如果线上线下数据模型都对上了，线下验证好线上效果还是差的话。很可能是某些小item被高估了

第三代解决方案架构-现在



第三代解决方案的经验

- 主要增加了召回模块
- 支持多条拉链的并行召回
- 支持离线或者在线更新拉链
- 支持灰度拉链热拔插

➢ **接入业务：**

- 信息流推荐，关联广告，搜索广告

第四代解决方案展望-2017年底

- **Spark，mllib的问题：**
  - Spark不支持FM，DNN等业界较先进的模型
  - Spark因为没有Parameter Server，executor的CPU利用率最多到30%
  - RDD的机制使得最慢的executor决定了迭代的速度
  - executor挂了之后重拉，持久化的块不会恢复

- **替代方案：**
  - CPU Cluster，通过Kubernetes+Docker弹性部署
  - GPU方案因为网络还不复杂，且数据量大。GPU利用率不足（40%）
  - 同时在考虑Angel作为过渡方案。


THANKS